PATENT ABSTRACTS OF JAPAN

(11)Publication number : 08-006805
(43)Dat f publication of application : 12.01.1996

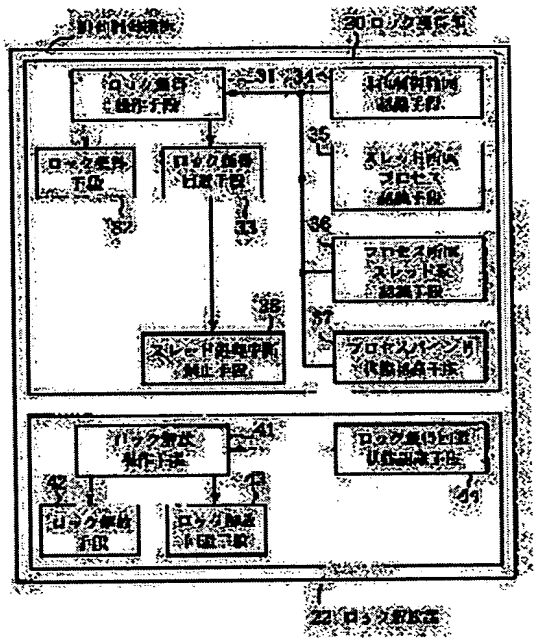
(51)Int.Cl. G06F 9/46
G06F 15/16

(21)Application number : 06-132878
(22)Date of filing : 15.06.1994

(71)Applicant : TOSHIBA CORP
(72)Inventor : YASUI KEISUKE

(54) MULTIPROCESSOR SYSTEM

(57)Abstract:
PURPOSE: To effectively utilize respectively processors under ptimum lock control.
CONSTITUTION: This multiprocessor system is equipped with an xclusive control range recognizing means 34 which recognizes whether or not data accessed by a thread are data shared only by threads in the same process, a thread belonging process recognizing m ans 35 which recognizes the process that the thread being xecuted belongs to, a process belonging thread quantity r cognizing means 36 which recognizes the number of threads belonging to the process that the thread being executes belongs to, and a lock acquiring operation means 31 which evades lock acquisition when recognizing that the process that the thread being executed belongs has only the thread in a critical access where the data shared only by threads in the same process are accessed on th basis of the recognition results of the respective recognizing m ans 34, 35, and 36.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the xaminer's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平8－6805

(43)公開日　平成８年(1996)１月12日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ	9/46	３６０　Ｄ	7737－5B		
	15/16	３４０　Ｚ			

審査請求　未請求　請求項の数３　ＯＬ　（全７頁）

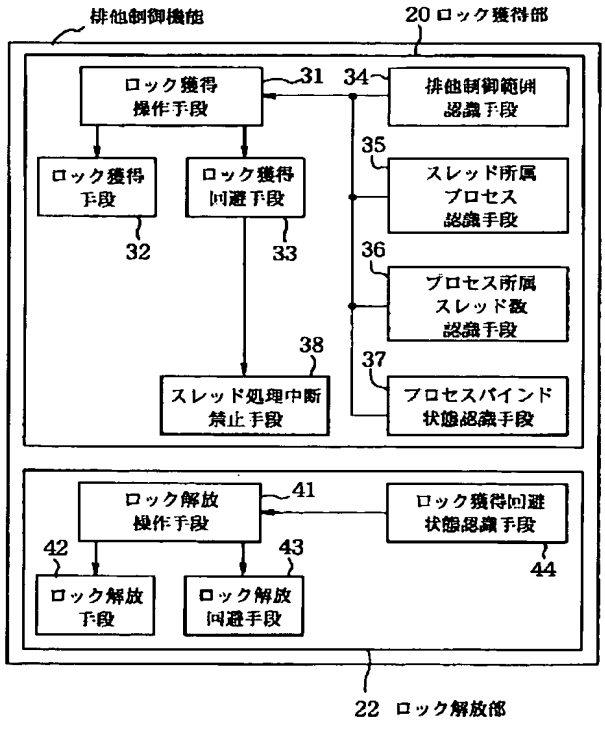
(21)出願番号　特願平6－132878

(22)出願日　平成６年(1994)６月15日

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地
(72)発明者　安井　啓介
東京都青梅市末広町２丁目９番地　株式会社東芝青梅工場内
(74)代理人　弁理士　鈴江　武彦

(54)【発明の名称】　マルチプロセッサシステム

(57)【要約】
【目的】最適なロック制御によって各プロセッサを有効利用することを可能にする。
【構成】スレッドによってアクセスされるデータが、同一プロセス内のスレッドのみによって共有されるデータであるか否かを認識する排他制御範囲認識手段３４と、実行中のスレッドが属するプロセスを認識するスレッド所属プロセス認識手段３５と、実行中のスレッドが属するプロセスに所属するスレッドの数を認識するプロセス所属スレッド数認識手段３６と、各認識手段３４，３５，３６による認識結果に基づいて、同一プロセス内のスレッドのみによって共有されるデータへアクセスするクリティカル領域で、実行中のスレッドが属するプロセスが、そのスレッド１つしか持たないことを認識した場合には、ロック獲得を回避させるロック獲得操作手段３１を具備して構成する。



1

【特許請求の範囲】

【請求項１】　１つのプロセス中に複数のスレッドを定義することが可能なマルチスレッド構造のオペレーティングシステムに基づいて動作するものであって、キャッシュを装備する複数のプロセッサ上で動作するプロセスによってデータが共有されるマルチプロセッサシステムにおいて、

スレッドによってアクセスされるデータが、同一プロセス内のスレッドのみによって共有されるデータであるか否かを認識する排他制御範囲認識手段と、

実行中のスレッドが属するプロセスを認識するスレッド所属プロセス認識手段と、

前記実行中のスレッドが属するプロセスに所属するスレッド数を認識するプロセス所属スレッド数認識手段と、

前記排他制御範囲認識手段、前記スレッド所属プロセス認識手段、及び前記プロセス所属スレッド数認識手段による認識結果に基づいて、同一プロセス内のスレッドのみによって共有されるデータへアクセスするクリティカル領域で、前記実行中のスレッドが属するプロセスが、そのスレッド１つしか持たないことを認識した場合には、ロック獲得を回避させるロック獲得操作手段と、

を具備したことを特徴とするマルチプロセッサシステム。

【請求項２】　同一プロセス内のスレッドのみによって共有されるデータへアクセスする時、前記実行中のスレッドが属するプロセスの持つ全てのスレッドが同じプロセッサにバインドされているかを認識するプロセスバインド状態認識手段をさらに具備し、

前記ロック獲得操作手段は、前記プロセスバインド状態認識手段による認識結果に基づいて、クリティカル領域の処理の間にスレッドの処理実行が中断されないことを保証した上で、ロック獲得を回避させることを特徴とする請求項１記載のマルチプロセッサシステム。

【請求項３】　前記クリティカル領域の処理が終了する際に、前記ロック獲得操作手段によってロック獲得が回避されている場合には、ロック解放を回避させるロック解放操作手段をさらに具備したことを特徴とする請求項１または請求項２記載のマルチプロセッサシステム。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、マルチスレッド構造のオペレーティングシステムによって動作するマルチプロセッサシステム、及び同システムにおける共有メモリの排他制御方法に関する。

【０００２】

【従来の技術】一般に、共有メモリを有し、各プロセッサにキャッシュが設けられているマルチプロセッサプロセッサシステムでは、複数のプロセッサによって共有される共有データにアクセスするクリティカル領域では、ロック操作などを行ない、権利を与えられたプロセスだけが、対象とするデータにアクセスできるようにする排他制御が行なわれている。マルチプロセッサシステムが、マルチスレッド構造のオペレーティングシステムによって動作するのであれば、スレッドに対してアクセスの権利が与えられる。

2

【０００３】また、この種、排他制御では、共有データが変更されると、変更されたことが他のプロセッサに通知される。共有データの変更が通知されたプロセッサは、同プロセッサが持つキャッシュに同じデータが保持されていれば、そのデータを無効化あるいは更新して、データの一貫性がとられるようにしている。

【０００４】一般に使用される排他制御の方法としてロック操作がある。ロック操作では、ある共有データに対して共有メモリ上にロック変数を用意し、共有データを処理する前にロック変数を確保（ロック変数にフラグを立てたり、所有者名を格納する）し、共有データをアクセスした後でロック変数をクリアする。

【０００５】プロセス（またはスレッド）は、ロック変数を確保しようとする際に、既に他のプロセス（またはスレッド）が、そのロック変数を確保している場合には、そのロック変数がクリアされるまで待ち状態となり、クリアされると直ちにロック変数を確保する。これによって、２つのプロセッサが同時に同じデータにアクセスして、データが不整合になることを防ぐことができる。

【０００６】ロック操作というのはマルチプロセッサの間での排他制御を行なうため、各プロセッサが持つキャッシュ上のデータについて更新等を行なうのみでなく、共有メモリ上に確保されたロック変数等のデータの更新を行なわなければならない。

【０００７】しかし、マルチプロセッサシステムにおいて、共有メモリ上のデータの更新に要する時間は、キャッシュ上のデータの更新に比べて非常に多く要する。従って、ロック操作を少なくすることができれば処理時間が短縮され、プロセッサを有効利用することができる。

【０００８】特にマルチスレッド構造のオペレーティングシステムによって動作するマルチプロセッサシステムでは、同一プロセス内のスレッドのみによって共有されるデータにアクセスする時に、そのスレッドが属するプロセスが持つスレッドが自分のみである場合のロック操作のような、不要なロック操作を避けることがプロセッサの有効利用につながる。

【０００９】

【発明が解決しようとする課題】このように従来のマルチスレッド構造のオペレーティングシステムによって動作するマルチプロセッサシステムにおいては、必要以上のロック操作が存在しているため、プロセッサが有効利用されないという問題があった。

【００１０】本発明は前記のような事情を考慮してなされたもので、最適なロック制御によって各プロセッサを

有効利用することが可能なマルチプロセッサシステムを提供することを目的とする。

【００１１】

【課題を解決するための手段】本発明は、１つのプロセス中に複数のスレッドを定義することが可能なマルチスレッド構造のオペレーティングシステムに基づいて動作するものであって、キャッシュを装備する複数のプロセッサ上で動作するプロセスによってデータが共有されるマルチプロセッサシステムにおいて、スレッドによってアクセスされるデータが、同一プロセス内のスレッドのみによって共有されるデータであるか否かを認識する排他制御範囲認識手段と、実行中のスレッドが属するプロセスを認識するスレッド所属プロセス認識手段と、前記実行中のスレッドが属するプロセスに所属するスレッドの数を認識するプロセス所属スレッド数認識手段と、前記排他制御範囲認識手段、前記スレッド所属プロセス認識手段、及び前記プロセス所属スレッド数認識手段による認識結果に基づいて、同一プロセス内のスレッドのみによって共有されるデータへアクセスするクリティカル領域で、前記実行中のスレッドが属するプロセスが、そのスレッド１つしか持たないことを認識した場合には、ロック獲得を回避させるロック獲得操作手段とを具備したことを特徴とする。

【００１２】また、同一プロセス内のスレッドのみによって共有されるデータへアクセスする時、前記実行中のスレッドが属するプロセスの持つ全てのスレッドが同じプロセッサにバインドされているかを認識するプロセスバインド状態認識手段をさらに具備し、前記ロック獲得操作手段は、前記プロセスバインド状態認識手段による認識結果に基づいて、クリティカル領域の処理の間にスレッドの処理実行が中断されないことを保証した上で、ロック獲得を回避させることを特徴とする。

【００１３】また、前記クリティカル領域の処理が終了する際に、前記ロック獲得操作手段によってロック獲得が回避されている場合には、ロック解放を回避させるロック解放操作手段をさらに具備したことを特徴とする。

【００１４】

【作用】このような構成によれば、共有データにアクセスするクリティカル領域を排他制御する際に、プロセスとスレッド数の関係、プロセッサとプロセスのバインド状態を考慮して、ロック操作を行なうことによって、プロセッサを有効利用することができる。

【００１５】つまり、同一プロセス内のスレッドのみによって共有されるデータへアクセスする時に、実行中のスレッドが属するプロセス中に含まれるスレッド数が１つである場合、また、同一プロセス内のスレッドのみによって共有されるデータへアクセスする時に、実行中のスレッドが属するプロセスの持つ全てのスレッドが同じプロセッサにバインドされている場合には、共有データに対してロックを獲得しなくても問題ないものとしてロック獲得処理を省略する。

【００１６】

【実施例】以下、図面を参照して本発明の一実施例を説明する。図１は本発明の一実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図である。図１に示すように、マルチプロセッサシステムは、複数のプロセッサ１０－１，１０－２，…，１０－ｎが設けられている。各プロセッサ１０－１，１０－２，…，１０－ｎは、それぞれキャッシュ１２－１，１２－２，…，１２－ｎを装備している。プロセッサ１０－１，１０－２，…，１０－ｎは、バス１４を介して共有メモリ１６と接続される。

【００１７】共有メモリ１６は、各プロセッサ１０－１，１０－２，…，１０－ｎが動作するためのマルチスレッド構造のオペレーティングシステムの他、各種データを格納している。また、共有メモリ１６には、各プロセッサ１０－１，１０－２，…，１０－ｎが、共有データをアクセスする際のロック制御に用いられるロック変数が格納される。

【００１８】各プロセッサ１０－１，１０－２，…，１０－ｎは、共有メモリ１６に格納されたプログラムに基づいて実現されるものであって、共有メモリ１６に格納されたデータに対する排他制御を行なう排他制御機能１８－１，１８－２，…，１８－ｎを有している。共有メモリ１６には、排他制御機能１８－１，１８－２，…，１８－ｎによって参照されるロック制御情報１９（詳細については後述する）が格納されている。

【００１９】排他制御機能１８－１，１８－２，…，１８－ｎは、同一プロセス内のスレッドのみによって共有されるデータへアクセスするクリティカル領域において、ロック操作を行なわなくても障害等が発生しないような状況にはロック操作を回避するように機能するものである。

【００２０】図２は本発明による排他制御方法を実現する排他制御機能１８－１，１８－２，…，１８－ｎの構成を示す機能ブロック図である。図２に示すように、本発明の排他制御機能１８－１，１８－２，…，１８－ｎは、ロック獲得操作を行なうロック獲得部２０、及びロック解放操作を行なうロック解放部２２から構成される。

【００２１】さらに、ロック獲得部２０は、ロック獲得操作手段３１、ロック獲得手段３２、ロック獲得回避手段３３、排他制御範囲認識手段３４、スレッド所属プロセス認識手段３５、プロセス所属スレッド数認識手段３６、プロセスバインド状態認識手段３７、及びスレッド処理中断禁止手段３８によって構成されている。

【００２２】また、ロック解放部２２は、ロック解放操作手段４１、ロック解放手段４２、ロック解放回避手段４３、及びロック獲得回避状態認識手段４４によって構成されている。

5

【００２３】ロック獲得操作手段３１は、排他制御範囲認識手段３４、スレッド所属プロセス認識手段３５、プロセス所属スレッド数認識手段３６、及びプロセスバインド状態認識手段３７による各認識結果に基づいて、ロック獲得手段３２とロック獲得回避手段３３を使い分けて、クリティカル領域を排他制御するためのロック獲得処理を実行する。すなわち、ロック獲得操作手段３１は、プロセスとスレッド数の関係や、プロセッサとプロセスのバインド状態を考慮したロック操作を行なう。

【００２４】ロック獲得手段３２は、共有データをアクセスする際に、ロック獲得操作手段３１によってロック獲得すべきものとされた場合に、ロックを獲得するものである。

【００２５】ロック獲得回避手段３３は、共有データをアクセスする際に、ロック獲得操作手段３１によってロック獲得の必要がないとされた場合にロック獲得回避処理を行なう。また、ロック獲得回避手段３３は、ロック獲得を回避する際に、スレッド処理中断禁止手段３８によって他のスレッドへ実行権を切り替えることを禁止させる。

【００２６】排他制御範囲認識手段３４は、共有データが、同じプロセス中のスレッドのみによって共有（アクセス）されるのか（プロセスローカル）、あるいは異なるプロセス中のスレッドでアクセスされるのか（プロセスグローバル）を、ロック制御情報１９のロック構造体（後述する）を参照して認識するものである。

【００２７】スレッド所属プロセス認識手段３５は、実行中のスレッドが所属するプロセスを、ロック制御情報１９のスレッド構造体（後述する）を参照して認識するものである。

【００２８】プロセス所属スレッド数認識手段３６は、実行中のスレッドが属するプロセスに所属するスレッドの数を、ロック制御情報１９のプロセス構造体（後述する）を参照して認識するものである。

【００２９】プロセスバインド状態認識手段３７は、実行中のスレッドが属するプロセスの持つ全てのスレッドが同じプロセッサにバインドされているかを、ロック制御情報１９のプロセス構造体（後述する）を参照して認識するものである。

【００３０】スレッド処理中断禁止手段３８は、ロック獲得を回避する場合に、他のスレッドへ実行権を切り替えることを禁止する指示を行なうものである。スレッド処理中断禁止手段３８は、システム側からの処理の中断要求に対しても、これを拒否する。

【００３１】ロック解放操作手段４１は、ロック獲得回避状態認識手段４４による認識結果に基づいて、ロック解放手段４２、及びロック解放回避手段４３を使い分けて、ロック獲得部２０によるロック獲得操作に対応するロック解放操作を行なうものである。

【００３２】ロック解放手段４２は、ロック獲得部２０によってロックが獲得されている場合に、クリティカル領域の処理が終了した後に、ロック解放処理を行なう。ロック解放回避手段４３は、ロック獲得部２０によってロック獲得が回避されている場合に、ロック解放回避処理を行なう。

6

【００３３】ロック獲得回避状態認識手段４４は、ロック制御情報１９のプロセッサ構造体（後述する）を参照して、ロック制御部２０によってロックの獲得の回避がなされているかを認識するものである。

【００３４】次に、共有メモリ１６に格納されるロック制御情報１９について説明する。ロック制御情報１９は、共有メモリ１６中に存在するが、排他制御機能１８－１，１８－２，…，１８－ｎによって参照される際には、対応するプロセッサ１０－１，１０－２，…，１０－ｎのキャッシュ１２－１，１２－２，…，１２－ｎにも格納される。

【００３５】ロック制御情報１９は、図３に示すように、ロック構造体５０、プロセス構造体５１、スレッド構造体５２、及びプロセッサ構造体５３に関する情報が格納されている。

【００３６】ロックのためのデータが格納されたロック構造体５０には、排他制御範囲２６に関する情報が含まれている。排他制御範囲２６は、ロックが保護する共有データが、あるプロセス内のスレッドのみにアクセスされるか、すなわち同じプロセス中のスレッド間で共有データにアクセスするプロセスローカルであるか、異なるプロセス中のスレッド間で共有データにアクセスするプロセスグローバルであるかを示す情報が設定されている。排他制御範囲２６に関する情報は、ロック構造体５０の初期化時に設定される。各スレッドが扱うデータは、対象が決まっているので、各スレッドの関係（プロセスグローバル、プロセスローカル）が初期化時に予め設定される。

【００３７】プロセスに関するデータが格納されているプロセス構造体５１には、プロセスのスレッド数５５及びバインド状態５６に関する情報が含まれている。スレッド数５５は、プロセスが持つスレッド数であり、バインド状態２９は、プロセスのプロセッサへのバインド状態を示している。

【００３８】スレッドに関するデータが格納されているスレッド構造体５２には、スレッドの所属プロセス５７を示す情報が含まれている。プロセッサに関するデータが格納されるプロセッサ構造体５３には、プロセッサに対するロック回避状態５８及び実行中断禁止状態５９に関する情報が含まれている。ロック回避状態５８は、プロセッサが、ロック獲得を回避している状態であるか否かを示し、実行中断禁止状態５９は、ロック操作を回避している場合にクリティカル領域の処理中には他のスレッドに処理が切り替わらないように、スレッドの処理の中断を制限することを示している。

【００３９】次に、本実施例の動作について説明する。まず、ロック獲得操作の処理について、図４に示すフローチャートを参照しながら説明する。まず、共有メモリ１６に格納された共有データに対してアクセスするためにロック獲得操作を実行する際、ロック獲得部２０の排他制御範囲認識手段３４は、ロック制御情報１９のロック構造体５０の排他制御範囲５４の情報を参照して、排他制御範囲の確認処理を行なう（ステップＡ１）。すなわち、プロセスグローバルであるかプロセスローカルであるかを認識する。

【００４０】ここで、共有データがローカルデータであって、同共有データにアクセスしようとするスレッドを含むプロセス中の他のスレッドのみにアクセスされると認識された場合には、スレッド所属プロセス認識手段３５は、実行中のスレッドが属するプロセスの持つスレッド数を、プロセス構造体５１のスレッド数５５を参照して認識する（ステップＡ２）。

【００４１】この結果、プロセスの持つスレッド数が複数であった場合、プロセスバインド状態認識手段３７は、実行中のスレッドが所属するプロセスのバインド状態を、プロセス構造体５１のバインド状態を参照して認識する（ステップＡ３）。

【００４２】プロセスが実行中のプロセッサにバインドされていない場合には、ロック獲得操作手段３１は、ロック獲得手段３２を用いて、すぐにロック獲得処理を行なう（ステップＡ４）。

【００４３】また、プロセスが実行中のプロセッサにバインドされている場合、すなわちプロセス中のプロセスローカルの関係にある全てのスレッドが同じプロセッサにバインドされている場合には、ロック獲得操作手段３１は、ロックの必要がないものとして、ロック獲得回避手段３３を用いて実行中断禁止処理を実行する。すなわち、スレッド処理中断禁止手段３８によって、プロセッサ構造体５３の実行中断禁止状態５９に実行中断禁止を示す情報をセットし、他のスレッドへ実行権を切り替えることを禁止する。

【００４４】他のスレッドへ実行権を切り替える実行中断禁止処理は、例えばＴＳＳ（タイムシェアリングシステム）等のために、強制的にシステムによって処理が中断されるような状況であっても、排他制御機能（処理中断禁止手段３８）のレベルで実行中断を禁止する。

【００４５】次に、ロック獲得操作手段３１は、ロック獲得回避手段３３を用いて、ロック獲得回避処理を実行する（ステップＡ６）。すなわち、プロセッサ構造体５３のロック回避状態５８に、ロック獲得回避を示す情報をセットする。

【００４６】なお、ステップＡ２において、スレッド所属プロセス認識手段３５が、実行中のスレッドが属するプロセスの持つスレッド数を、プロセス構造体５１のスレッド数５５を参照して認識した結果、プロセス中に１つのスレッドしかない場合には、ロック獲得操作手段３１は、ロックの必要がないものとして、ロック獲得回避手段３３を用いて、ロック獲得回避処理を実行する（ステップＡ６）。すなわち、プロセッサ構造体５３のロック回避状態５８に、ロック獲得回避を示す情報をセットする。

【００４７】一方、ステップＡ１において、ロックが保護する共有データが、他のプロセスからもアクセスされるグローバルデータであると認識された場合には、ロック獲得操作手段３１は、ロック獲得手段３２を用いて、すぐにロック獲得処理を行なう（ステップＡ４）。

【００４８】こうして、ステップＡ４におけるロック獲得処理、またはステップＡ６におけるロック獲得回避処理（ロックを獲得せずに）が完了すると、クリティカル領域の処理に移る。

【００４９】ロック獲得部２０における処理では、共有メモリ１６に格納された共有データにアクセスする際に、ロック獲得を行なわなくても問題がない場合にはロック獲得処理を省いている。また、クリティカル領域の排他制御を行なう際には、ロック制御情報１９がキャッシュ上に存在しているので、実行中断禁止処理、ロック獲得回避処理は、高速に実行することができる。

【００５０】次に、ロック解放操作の処理について、図５に示すフローチャートを参照しながら説明する。まず、クリティカル領域の処理が終了したところで、ロック解放操作を実行する際、ロック解放部２２のロック獲得回避状態認識手段４４は、ロック制御情報１９のプロセッサ構造体５３のロック回避状態５８の情報を参照して、ロック回避状態の確認処理を行なう（ステップＢ１）。すなわち、ロック回避状態５８に、ロック獲得回避を示す情報がセットされているか否かを認識する。

【００５１】ここで、ロック獲得状態にあると認識された場合には、ロック解放操作手段４１は、ロック解放手段４２を用いてロック解放処理を行ない、排他制御処理を終了する（ステップＢ４）。

【００５２】また、ロック回避状態にあると認識された場合には、ロック解放操作手段４１は、ロック解放回避手段４３を用いて、ロック回避状態解除処理を行なう。すなわち、ロック解放回避手段４３は、ロック制御情報１９のプロセッサ構造体５３のロック回避状態３３をリセットする（ステップＢ２）。

【００５３】さらに、ロック解放操作手段４１は、ロック解放回避手段４３を用いて、実行中断禁止解除処理を行なう。すなわち、ロック解放回避手段４３は、ロック制御情報１９のプロセッサ構造体５３の実行中断禁止状態５９をリセットし、排他制御処理を終了する（ステップＢ３）。すなわち、共有データに対するロック制御を回避している場合には、クリティカル領域の処理を出る時には、ロックの解放処理を行なわない。

【００５４】なお、前記実施例では、単純なクリティカ

ル領域に対する排他制御について説明しているが、例えばロックが階層的に行なわれるような処理については次のようにすることもできる。

【００５５】ロック獲得が必要となった場合、ロック獲得操作手段３１は、ロック獲得手段３２を用いて、ロック獲得処理を実行する。この際、ロック制御情報１９のプロセッサ構造体５３の実行中断禁止状態を、ロックの階層が上がる毎に順次、階層のレベルに応じて更新する。

【００５６】そして、各階層レベルのクリティカル領域の処理が終了する毎に、ロック解放操作手段４１は、ロック解放手段４２を用いて、階層毎に順次ロックを解放していく。

【００５７】また、ロック獲得回避処理が階層的にされている場合には、同様にしてロック回避状態解除処理を、階層毎に順次行なう。このようにして、同一プロセス内のスレッドのみによって共有されるデータへアクセスするクリティカル領域で、前記実行中のスレッドが属するプロセスが、そのスレッド１つしか持たない場合、あるいは実行中のスレッドが属するプロセスの持つ全てのスレッドが同じプロセッサにバインドされている場合には、ロック操作を行なわないことにより、プロセッサを有効に利用することができる。

【００５８】特に、共有メモリ１６に対する書き込み（ロック変数に対する）に時間がかかったり、ロック操作に伴い他のプロセッサに対する通知等に多くのコストを要するシステムにおいては、ロック操作を省くことで多くの効果が得られる。

【００５９】

【発明の効果】以上詳述したように本発明によれば、プロセスとスレッド数の関係、プロセッサとプロセスのバインド状態を考慮して、不要なロック制御を省くことによって、最適なロック制御によって各プロセッサを有効利用することが可能となるものである。

【図面の簡単な説明】

【図１】本発明の一実施例に係わるマルチプロセッサシステムの概略構成を示すブロック図。

【図２】本発明による排他制御方法を実現する排他制御機能１８－１，１８－２，…，１８－ｎの構成を示す機能ブロック図。

【図３】本実施例におけるロック制御情報１９に含まれる各種情報の一例を示す図。

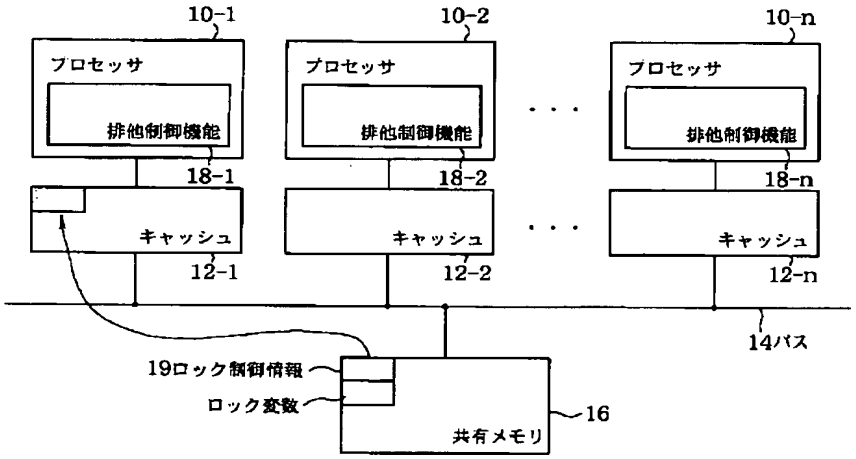
【図４】本実施例におけるロック獲得操作の処理を説明するためのフローチャート。

【図５】本実施例におけるロック解放操作の処理を説明するためのフローチャート。

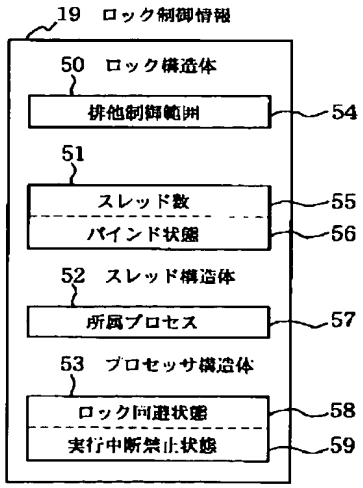
【符号の説明】

１０－１，１０－２，…，１０－ｎ…プロセッサ、１２－１，１２－２，…，１２－ｎ…キャッシュ、１４…バス、１６…共有メモリ、１８－１，１８－２，…，１８－ｎ…排他制御機能、１９…ロック制御情報、２０…ロック獲得部、２２…ロック解放部、３１…ロック獲得操作手段、３２…ロック獲得手段、３３…ロック獲得回避手段、３４…排他制御範囲認識手段、３５…スレッド所属プロセス認識手段、３６…プロセス所属スレッド数認識手段、３７…プロセスバインド状態認識手段、３８…スレッド処理中断禁止手段、４１…ロック解放操作手段、４２…ロック解放手段、４３…ロック解放回避手段、４４…ロック獲得回避状態認識手段。

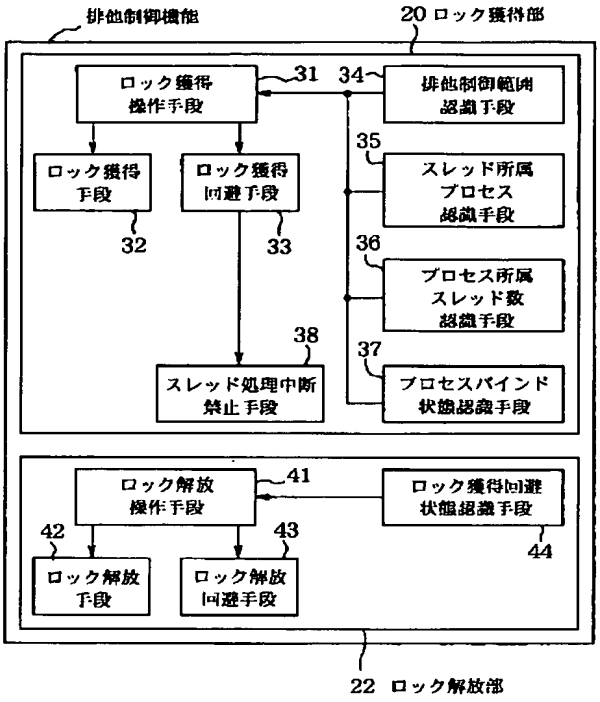
【図１】



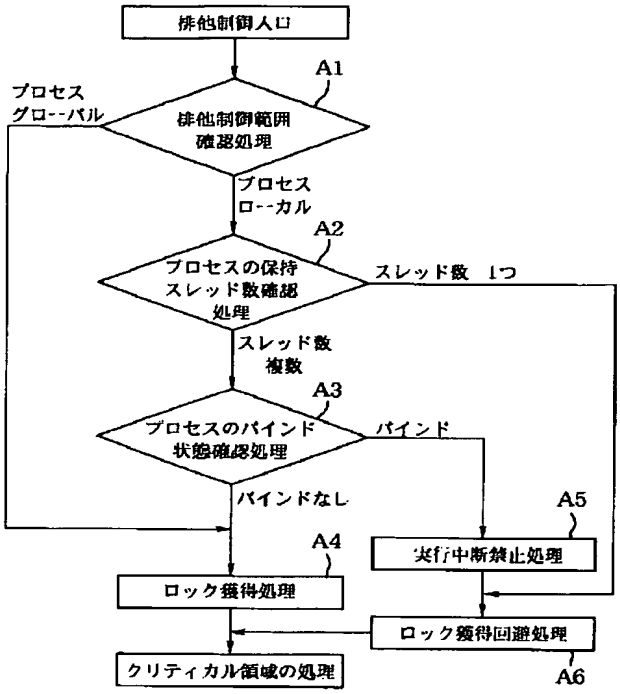
【図３】



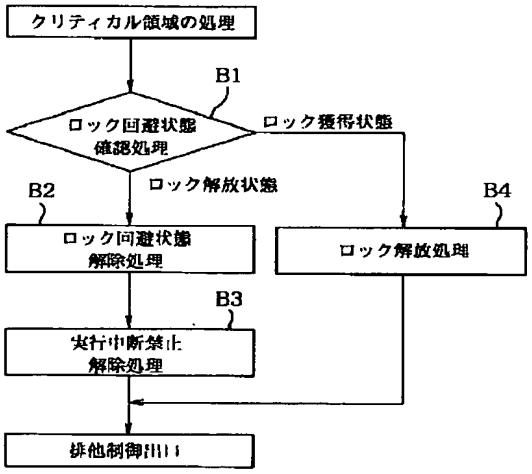
【図２】


排他制御機能
20 ロック獲得部
ロック獲得操作手段
31
34
排他制御範囲認識手段
ロック獲得手段
32
ロック獲得回避手段
33
35
スレッド所属プロセス認識手段
36
プロセス所属スレッド数認識手段
38
スレッド処理中断禁止手段
37
プロセスバインド状態認識手段
ロック解放操作手段
41
ロック獲得回避状態認識手段
42
44
43
ロック解放手段
ロック解放回避手段
22 ロック解放部

【図４】


排他制御入口
A1
プロセスグローバル
排他制御範囲確認処理
プロセスローカル
A2
プロセスの保持スレッド数確認処理
スレッド数　1つ
スレッド数複数
A3
プロセスのバインド状態確認処理
バインド
バインドなし
A5
実行中断禁止処理
A4
ロック獲得処理
ロック獲得回避処理
クリティカル領域の処理
A6

【図５】


クリティカル領域の処理
B1
ロック回避状態確認処理
ロック獲得状態
ロック解放状態
B2
B4
ロック回避状態解除処理
ロック解放処理
B3
実行中断禁止解除処理
排他制御出口